

AMENDMENTS TO THE CLAIMS

Claims 1-3, 5-11, 16, 18-20, 26-29, and 35-44 were pending at the time of the Office Action.

Claims 12-15, 21-25, and 30-34 were withdrawn from consideration.

Claims 4 and 17 were previously canceled.

Claims 5, 28, and 36 are hereby canceled.

Claims 1, 6-7, 26-27, 35, 37, 41, and 43 are amended.

Claims 1-3, 6-11, 16, 18-20, 26-27, 29, 35, and 37-44 remain pending.

1. (Currently Amended) A method for managing access to resources, comprising:

generating a list of resource signatures, each of the resource signatures being generated based at least on a plurality of function names included in an import table of a corresponding resource;

accessing the list of resource signatures, each of the resource signatures configured with an accessibility status, wherein the accessibility status includes one of loadable and restricted;

generating a verification signature for a requested resource, wherein
~~generating a verification signature includes: the verification signature being generated based at least on a plurality of function names included in an import table of the requested resource~~

retrieving a plurality of names from the import table, wherein the plurality of names include function names;

sorting the retrieved names;

concatenating the sorted names; and

executing a cryptographic manipulation of the concatenated names;
comparing the verification signature for the requested resource to the
list of resource signatures;
executing the requested resource if the resource signature matches the
verification signature and the accessibility status is loadable; and
preventing the requested resource from execution if the resource
signature matches the verification signature and the accessibility
status is restricted.

2. (Original) A method according to Claim 1, wherein the resources include applications or programs.

3. (Previously Presented) A method according to Claim 1, wherein each of the resource signature is generated based further on one or more dynamic link library (DLL) names, and wherein the verification signature is generated based further on one or more dynamic link library (DLL) names.

4. (Canceled).

5. (Canceled).

6. (Currently Amended) A method according to Claim 1~~-Claim 5~~, wherein executing a cryptographic manipulation includes executing a hash function.

7. (Currently Amended) A method according to Claim 1~~-Claim 5~~, wherein the plurality of names further include DLL names.

8. (Previously Presented) A method according to Claim 1, further comprising preventing the requested resource from execution if the resource signature does not match the verification signature and the accessibility status is restricted.

9. (Previously Presented) A method according to Claim 1, further comprising preventing the requested resource from execution if the resource signature does not match the verification signature and the accessibility status is loadable.

10. (Original) A method according to Claim 1, wherein a same procedure is followed to generate each of the resource signatures and to generate a verification signature.

11. (Previously Presented) A method according to Claim 1, further comprising coding the generated list of resource signatures into a dynamic link library (DLL) of an operating system kernel.

12. (Withdrawn) A method for generating an application identifier, comprising:

- receiving a command to generate an identifier for an application;
- retrieving an import table from the application;
- sorting information from the retrieved import table; and
- performing a cryptographic function on the sorted information.

13. (Withdrawn) A method according to Claim 12, wherein the import table is retrieved from an executable of the application.

14. (Withdrawn) A method according to Claim 12, wherein sorting information from the retrieved import table includes sorting function names according to at least one predetermined criterion.

15. (Withdrawn) A method according to Claim 12, wherein performing a cryptographic function on the sorted information includes performing a one-way hash function.

16. (Previously Presented) A method of restricting particular applications, comprising:

receiving a list of application fingerprints corresponding respectively to restricted applications;

receiving a request to execute an application;

generating a confirmation fingerprint for the requested application,

wherein generating the fingerprint includes:

retrieving a plurality of names from an import table, wherein the plurality of names include function names;

sorting the retrieved names;

concatenating the sorted names in a predetermined manner; and

hashing the concatenated names;

comparing the confirmation fingerprint to the list of application fingerprints; and

restricting the requested application if the confirmation fingerprint matches one of the application fingerprints respectively corresponding to restricted applications.

17. (Canceled).

18. (Previously Presented) A method according to Claim 16, wherein the confirmation fingerprint is further generated from DLL names included in the import table of the requested application, and wherein retrieving a plurality of names further includes retrieving DLL names.

19. (Previously Presented) A method according to Claim 16, wherein hashing the organized names include performing one of a Message Digest (MD) 5 hash and a Secure Hash Algorithm (SHA) 1.

20. (Original) A method according to Claim 16, wherein the restricted applications are not licensed.

21. (Withdrawn) An apparatus, comprising:

- a licensing manager component to provide identification for an application and to further assign a classification of the application as being restricted or unrestricted; and
- a developer component to code an operating system to include the identification in correspondence with the classification.

22. (Withdrawn) An apparatus according to Claim 21, wherein the licensing manager component is to generate identification for an application using an import table from the application.

23. (Withdrawn) An apparatus according to Claim 21, wherein the licensing manager component is to provide a restricted classification for an unlicensed application.

24. (Withdrawn) An apparatus according to Claim 23, wherein, for an unlicensed application, the licensing manager component is to:

- retrieve an import table from an executable of the application;
- sort and string together function identifiers from the import table; and
- execute a cryptographic algorithm on the function identifiers.

25. (Withdrawn) An apparatus according to Claim 24, wherein the cryptographic algorithm is a hashing algorithm.

26. (Currently Amended) An apparatus, comprising:

- an interface to receive a request for a running state of an application;
- an application identifier to generate an application digital signature for the application, the application identifier is to: based a least on a plurality of function names included in an import table of the application;

- retrieve a plurality of names from an import table from the application,

- wherein the plurality of names include function names;

- sort the retrieved names;

- concatenate the sorted names; and

- execute a cryptographic manipulation of the concatenated names;

- an application manager to match the application digital signature against a list of stored digital signatures indicating whether corresponding applications are eligible or ineligible for a running state; and
- an enabler to enable the running state for the application if the application digital signature is not matched to a stored digital signature indicating that the application is ineligible.

27. (Currently Amended) An apparatus according to Claim 26, wherein the application identifier is to execute a cryptographic manipulation by execute a hash function on the concatenated names ~~generate an application digital signature using an import table from the application.~~

28. (Canceled).

29. (Previously Presented) An apparatus according to Claim 28, wherein the instruction to hash further includes an instruction to execute an MD5 hashing algorithm on the stringed function names.

30. (Withdrawn) A computer-accessible medium having one or more instructions that are executable by one or more processors, the one or more instructions causing the one or more processors to:

- receive a command to generate a program fingerprint;
- sort static data from within the program; and
- create a program fingerprint using the sorted static data.

31. (Withdrawn) A computer-accessible medium according to Claim 30, wherein the static data includes an import table.

32. (Withdrawn) A computer-accessible medium according to Claim 30, wherein the static data includes function names corresponding to an import table from an executable of the program.

33. (Withdrawn) A computer-accessible medium according to Claim 30, wherein the one or more instructions that cause the one or more processors to sort static data further cause the one or more processors to:

organize the function names from an import table corresponding to a program executable in accordance with at least one predetermined criterion; and
link the organized function names.

34. (Withdrawn) A computer-accessible medium according to Claim 33, wherein the one or more instructions that cause the one or more processors to create a program fingerprint further cause the one or more processors to execute a hashing algorithm on the sorted static data.

35. (Currently Amended) A computer-accessible storage medium having an application programming interface (API), the API having one or more instructions to cause one or more processors to:

receive a request to run a program;

generate a digital signature for the program-based ~~at least on a plurality of function names included in an import table of the program by~~
causing the one or more processors to:

retrieve a plurality of names from an import table of the program,

wherein the plurality of names include function names;

sort the retrieved names;

concatenate the sorted names; and

execute a cryptographic manipulation of the concatenated names;

compare the generated digital signature against a compilation of digital signatures corresponding to restricted programs; and

enable only those programs for which the signature does not match with any of the compiled signatures.

36. (Canceled).

37. (Currently Amended) A computer-accessible storage medium according to ~~Claim 35~~ ~~Claim 36~~, wherein the one or more instructions to cause the one or more processors to execute a cryptographic manipulation includes causing the one or more processors to execute a hash function on the concatenated names ~~to generate a digital signature for the program cause the one or more processors to hash a sorted list of function names from the import table.~~

38. (Previously Presented) A computer-accessible storage medium according to Claim 35, wherein the API is included in an operating system.

39. (Previously Presented) A computer-accessible storage medium according to Claim 35, wherein the operating system runs on a web server.

40. (Previously Presented) A computer-accessible storage medium according to Claim 35, wherein the operating system runs on an application server.

41. (Currently Amended) A license enforcement method, comprising:

generating a digital signature for each of a plurality of applications
based at least on a plurality of function names included in an import
table of each application;

classifying each of the digital signatures in accordance with a licensing
status for the corresponding applications;

coding an operating system to:

include the classified digital signatures[[],];

generate a digital signature for a requested application ~~based at least on a plurality of function names included in an import table of the requested application~~ by perform steps that include:

retrieve a plurality of names from an import table of the program, wherein the plurality of names include function names;

sort the retrieved names;

concatenate the sorted names; and

execute a cryptographic manipulation of the concatenated names;[[,]]

compare the digital signature for the requested application to the classified digital signatures[[,]]; and

run the requested application when the digital signature for the requested application does not map to digital signature classified as not being licensed.

42. (Original) A license enforcement method according to Claim 41, further comprising downloading an updated list of the classified digital signatures to the operating system.

43. (Currently Amended) A method according to Claim 1-Claim 7, wherein executing a cryptographic manipulation includes executing a hash function.

44. (Previously Presented) A method according to Claim 41, wherein generating a digital signature for a requested application further includes generating the request digital signature based on a plurality of function names and

one or more dynamic link library (DLL) names included in an import table of the requested application.